LA-UR-00-422

Title: Parallel Image Processing with Autowaves: Segmentation and Edge Extraction

Authors: Alexei N. Skourikhine

Accepted to: 4^{th} *World Multiconference on Systemics, Cybernetics,* and Informatics, Orlando, Florida, July 23–26, 2000

Proc. of the 4th World Multiconference on Systemics, Cybernetics and Informatics, vol. 5, Image, Acoustic, Speech and Signal Processing: Part 1, pp. 302-305, ISBN 980-07-6691-X, Orlando, Florida, July 23-26, 2000.

Los Alamos
NATIONAL LABORATORY

Parallel Image Processing with Autowaves: Segmentation and Edge Extraction

Alexei N. Skourikhine
Safeguards Systems Group, MS E541
Los Alamos National Laboratory
Los Alamos, NM 87545, USA
alexei@lanl.gov

ABSTRACT

Biologically inspired image-processing algorithms like pulse-coupled neural networks find many applications in image preprocessing, including segmentation and edge extraction. Two highly parallel methods of edge extraction from gray-scale images using a pulse-coupled neural network are presented. The approach of both methods is based on phenomenon of autowaves, whose properties enable efficient parallel image processing.

Keywords: Pulse Coupled Neural Network, Autowave, Image Processing, Segmentation, Edge Extraction

1. INTRODUCTION

The term *autowave* was first introduced by R.V. Khorlov in [1] to indicate "autonomous waves." Autowaves represent a particular class of nonlinear waves, which propagates in an active media at the expense of the energy stored in the medium. They are often encountered in many biological processes, e.g., propagation in nerve fibers. Autowaves possess some typical properties that are different from those of classical waves in conservative systems. The shape and amplitude of autowaves remain constant during propagation, they do not reflect from inhomogeneities, there is no interference because two colliding autowaves annihilate each other. Nonetheless, both autowaves and classical waves share the property of diffraction. These properties, especially annihilation and diffraction, make autowaves very useful for image analysis [2]. The purpose of this paper is to show how a pulse-coupled neural network (PCNN) producing autowaves can be used for image preprocessing applications, such as edge extraction.

2. IMAGE PROCESSING USING PULSE COUPLED NEURAL NETWORKS

The PCNN is a biologically inspired algorithm for image preprocessing [4,5]. It is to a very large extent based on the Eckhorn model of the cat visual cortex [3].

The typical neuron of the PCNN is shown in Fig. 1. The equations for a single iteration of the PCNN are

$$
\begin{aligned}
&F_{ij}[n] = e^{-\alpha_F} F_{ij}[n-1] + S_{ij} + V_F \sum_{kl} m_{ijkl} Y_{kl}[n-1] \\
&L_{ij}[n] = e^{-\alpha_L} L_{ij}[n-1] + V_L \sum_{kl} w_{ijkl} Y_{kl}[n-1] \\
&U_{ij}[n] = F_{ij}[n](1 + \beta L_{ij}[n]) \\
&Y_{ij}[n] = \begin{cases} 1, & if \;\; U_{ij}[n] > \Theta_{ij}[n] \\ 0, & otherwise \end{cases} \\
&\Theta_{ij}[n] = e^{-\alpha_\Theta} \Theta_{ij}[n-1] + V_\Theta Y_{ij}[n] \qquad (1)
\end{aligned}
$$

where S is the input signal, F is the feed, L is the link, U is the internal activity, Y is the pulse output, Θ is the dynamic threshold. The weight matrices M and W are local interconnections and β is the linking constant. All neurons' values are 0 at $n < 0$.

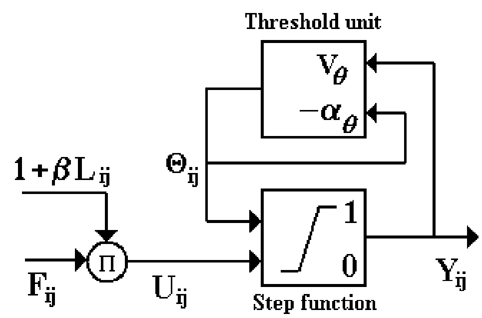

Fig.1. The basic PCNN neuron.

The basic simplified structure of the pulse-coupled neural network processor for a 2-D input image is shown in Fig. 2. An input gray-scale image is composed of $M \times N$ pixels. This image can be represented as an array of $M \times N$ normalized intensity values. Then the array is fed in at the $M \times N$ inputs of PCNN. Since initially all neurons are set to 0, the input results in activation of all of the neurons at a first iteration. The threshold of each neuron, Θ, significantly increases when the neuron fires, then the threshold value decays. When the threshold falls below the respective neuron's potential, U, the neuron fires again, which raises the threshold. The process continues creating binary pulses for each neuron. While this process goes on, neurons encourage their neighbors to fire simultaneously in a way that is supported through interconnections. The firing neurons begin to communicate with their nearest neighbors, which in turn communicate with their neighbors. The result is an autowave that expands from active regions. Thus, if a group of neurons is close to firing, then one neuron can trigger the group. As a result of linking between neurons, the pulsing activity of invoked neurons leads to the synchronization between groups of neurons corresponding to sub-regions of the image with similar properties and produces a temporal series of binary images. These phenomena of synchronization and autowaves support image segmentation and edge extraction.

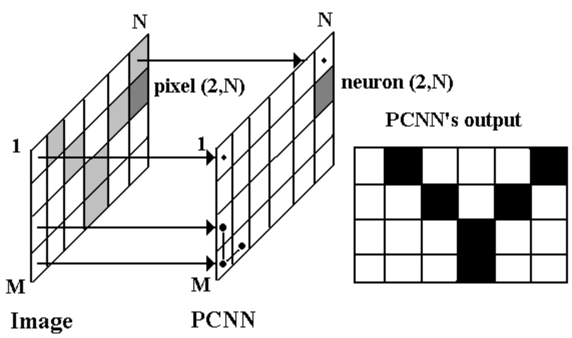

Fig. 2. Image processing using a pulse-coupled neural network (PCNN).

3. EDGE EXTRACTION

In one of the conventional methods of extracting an edge of an object existing in an image, an input image is binarized using a certain threshold value, and a binarizing borderline is extracted as an edge. In another methods, an edge is extracted using a differential operator without performing binarization. However, real images are noisy, and either of these methods will often not lead to the detection of acceptable edges. An algorithm that implements preliminary smoothing based on similarities of properties of adjacent pixels can improve edge extraction. Adjustable smoothing that is controlled by β, and binding of pixels based on their properties, such as intensities, are the inherent characteristics of the PCNN.

By utilizing the PCNN, the gray-scale image (Fig. 3) is processed by the PCNN algorithm to produce a binary image containing the segmentation result (Fig. 4). Then, edge extraction (Fig. 5) from the binary image can be done two ways, using the property of a travelling linking wave, L, which exhibits autowave properties.

One method is to then take the obtained binary image (Fig. 6), invert it (Fig. 7), and present this inverted image to a PCNN. The first output image from PCNN is an exact copy of the inverted image, and at the next iteration PCNN produces an image (Fig. 9) containing edges of the white blobs of the original binary image (Fig. 6). These are edges we are looking for. The method is based on the property of linking wave propagation from currently active regions of the image to neighbors that were inactive before. As a result, pixels close to the active regions become activated and produce the edges.

The other method is to wait until a segmented region (Fig. 6) has pulsed. Immediately after the pulse a linking wave is launched from the edge of the active region. In order to extract the edge of the region, we have to perform a logical “AND” of both the front of the linking wave (Fig. 8) and the original image (Fig. 6). The result of the operation is the edges of currently active regions (Fig. 9). In Figs. 10 and 11, details are shown highlighting the difference between the linking wave front and the correct edges. From their comparison, the difference between the wave front and the edges of the active (white) regions is clearly visible. Depending on our needs, we can easily extract edges from the same wave front either for an active region or for its complement (black region). Another modification of this method of edge extraction is to check the linking input value for each neuron of the region that has just pulsed. Comparison of this value against a neuron’s threshold, which depends on the interconnection weights, gives edge pixels because it will be always less than a neuron’s threshold for edge pixels.

Edges produced by PCNN can be of two types depending on the weight template that is used for the neuron neighborhood. One edge is of strictly one pixel wide (Fig. 11). Linking weights template to extract this type of edge is

$$W=\begin{bmatrix}0&1&0\\1&0&1\\0&1&0\end{bmatrix} \qquad (2)$$

The other type is an edge of variable width of one-two pixels that is influenced by geometry (Fig. 12). The corresponding linking weights template is

$$W=\begin{bmatrix}1&1&1\\1&0&1\\1&1&1\end{bmatrix} \qquad (3)$$

The result of segmentation is less noisy if the template (3) is used. Thus, two templates are actually used in the described approach. First, template (3) is used to perform a segmentation of the original image. Then, template (2) is used to process binary images to extract one pixel wide edges.

4. CONCLUSIONS

We described the image segmentation approach based on PCNN and presented two methods through which PCNN can be used to extract edges from images. In a coupled array of PCNN neurons, linking travelling waves make possible edge detection by propagation through the image. All the methods are inherently parallel. Other opportunities for the use of PCNN travelling waves include detection of closed curves, e.g. holes, within extracted blobs, image denoising and image thinning. Further study of the use of different weight templates similar to what is done in the cellular neural network field can open new, interesting options in the PCNN applications to image processing.

5. REFERENCES

[1] V. I. Krinsky, Ed., Self-Organization: Autowaves and Structures far from Equilibrium, New York: Springer-Verlag, 1984.

[2] L. Kuhnert, K. I. Agladze, V. I. Krinsky, “Image processing using light-sensitive chemical waves,” Nature, vol. 337, pp. 244–247, 1989.

[3] R. Eckhorn, H. J. Reitboeck, M. Arndt, P. Dicke, “Feature linking via synchronization among distributed assemblies: Simulations of results from cat visual cortex,” Neural Comp., 2, 293–307, 1990.

[4] J. L. Johnson, M. L. Padgett, “PCNN models and applications,” IEEE Trans. on Neural Networks, vol. 10, no. 3, pp. 480–498, 1999.

[5] T. Lindblad, J. Kinser, Image Processing using Pulse-Coupled Neural Networks, Springer-Verlag, 1998.

Fig. 3. The original gray-level image.

Fig. 4. PCNN output.

Fig. 5. Detected edges.

Fig. 6. Segmented image.

Fig. 7. Inverted segmented image.

Fig. 8. Linking wave.

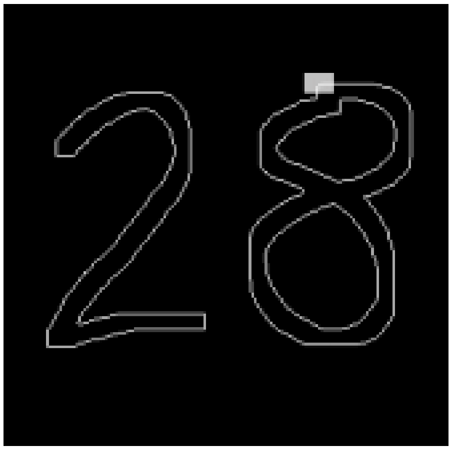

Fig. 9. Extracted edges.

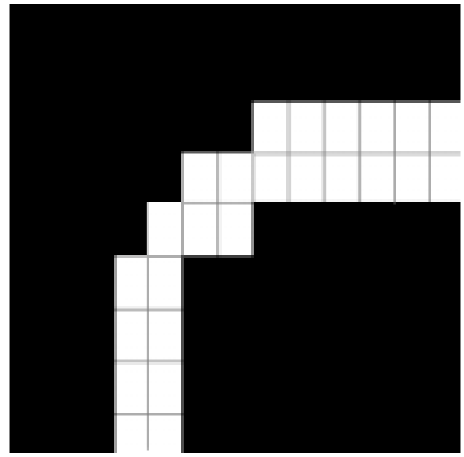

Fig. 10. Region highlighted in Fig. 8.

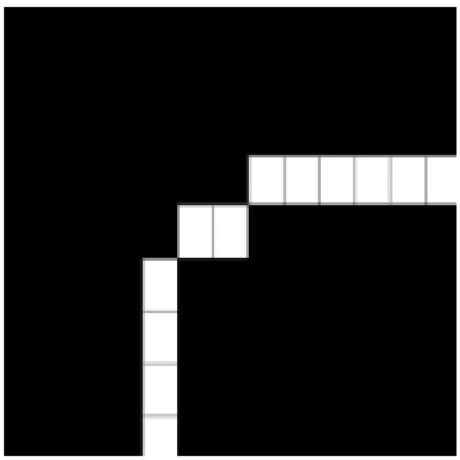

Fig. 11. Region highlighted in Fig. 9. Edge extracted using neighborhood consisting of 4 pixels.

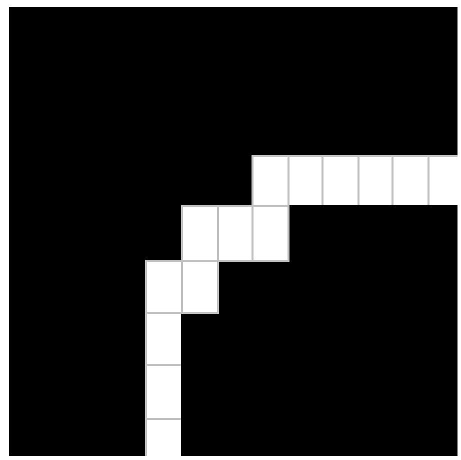

Fig. 12. Edge extracted using neighborhood consisting of 8 pixels.